Localização de Pontos

INF2604 – Geometria Computacional

Waldemar Celes
celes@inf.puc-rio.br

Departamento de Informática, PUC-Rio

Agenda

Localização de pontos

Par próximo

Localização de pontos

Localização de pontos

Teorema de Jordan

- Toda curva contínua fechada que não se cruza divide o plano em três regiões: interior, exterior, fronteira

L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Localização de pontos

Teorema de Jordan

- Toda curva contínua fechada que não se cruza divide o plano em três regiões: interior, exterior, fronteira

Problema

- Dado um *polígono simples* P e um ponto $\mathbf{p}$, localizar $\mathbf{p}$ em relação a P:
  - No interior, no exterior ou na fronteira

L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

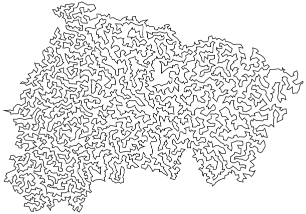
Localização de pontos

Teorema de Jordan

- Toda curva contínua fechada que não se cruza divide o plano em três regiões: interior, exterior, fronteira

Problema

- Dado um *polígono simples* P e um ponto $\mathbf{p}$, localizar $\mathbf{p}$ em relação a P:
  - No interior, no exterior ou na fronteira



L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

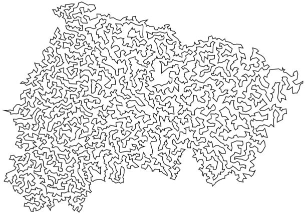
Localização de pontos

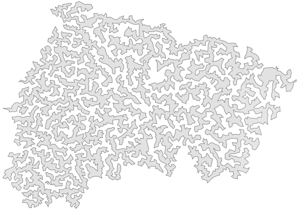
Teorema de Jordan

- Toda curva contínua fechada que não se cruza divide o plano em três regiões: interior, exterior, fronteira

Problema

- Dado um *polígono simples* P e um ponto $\mathbf{p}$, localizar $\mathbf{p}$ em relação a P:
  - No interior, no exterior ou na fronteira

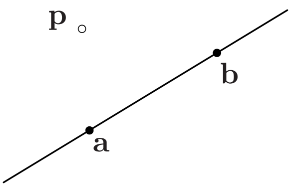




L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

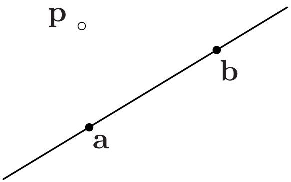
Localização de pontos

Localização de ponto em relação a
uma reta L definida pelos pontos $\mathbf{a}$ e $\mathbf{b}$



Localização de pontos

Localização de ponto em relação a
uma reta L definida pelos pontos $\mathbf{a}$ e $\mathbf{b}$

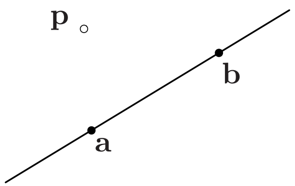


Reta L divide o plano em 2 semi-planos

- À esquerda
- À direita

Localização de pontos

Localização de ponto em relação a uma reta L definida pelos pontos $\mathbf{a}$ e $\mathbf{b}$



Localizar $\mathbf{p}$ em relação a L:

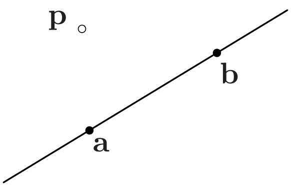
- No semi-plano à esquerda
- No semi-plano à direita
- Sobre L

Reta L divide o plano em 2 semi-planos

- À esquerda
- À direita

Localização de pontos

Localização de ponto em relação a
uma reta L definida pelos pontos $\mathbf{a}$ e $\mathbf{b}$



Reta L divide o plano em 2 semi-planos

- À esquerda
- À direita

Localizar $\mathbf{p}$ em relação a L:

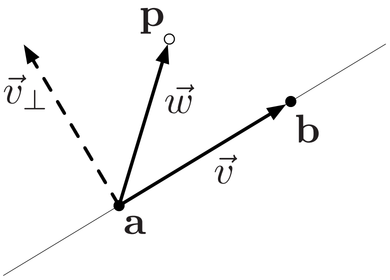
- No semi-plano à esquerda
- No semi-plano à direita
- Sobre L
  - Antes de $\mathbf{a}$
  - Em $\mathbf{a}$
  - Entre $\mathbf{a}$ e $\mathbf{b}$
  - Em $\mathbf{b}$
  - Depois de $\mathbf{b}$

Localização de ponto em relação à reta

Ponto $\mathbf{p}$ estará à esquerda de L sse:

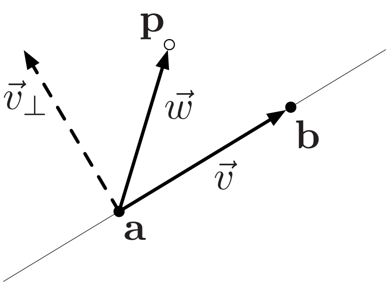
$$\vec{\mathbf{v}} = \mathbf{b} - \mathbf{a}$$
$$\vec{\mathbf{w}} = \mathbf{p} - \mathbf{a}$$
$$v_{\perp} = [-v_y, v_x]^T$$

$$v_{\perp}.\vec{\mathbf{w}} > 0$$



Localização de ponto em relação à reta

Ponto $\mathbf{p}$ estará à esquerda de L sse:



$$\vec{\mathbf{v}} = \mathbf{b} - \mathbf{a}$$
$$\vec{\mathbf{w}} = \mathbf{p} - \mathbf{a}$$
$$v_\perp = [-v_y, v_x]^T$$

$$v_\perp . \vec{\mathbf{w}} > 0$$

Ou:

$$orient < \mathbf{a}, \mathbf{b}, \mathbf{p} >= \begin{vmatrix} 1 & a_x & a_y \\ 1 & b_x & b_y \\ 1 & p_x & p_y \end{vmatrix} > 0$$

Localização de ponto em relação à reta

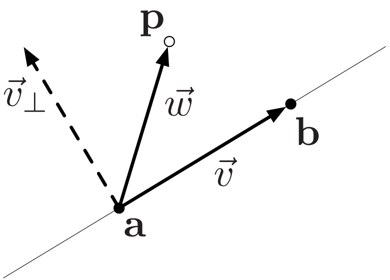
Ponto $\mathbf{p}$ estará à esquerda de L sse:

$$\vec{\mathbf{v}} = \mathbf{b} - \mathbf{a}$$

$$\vec{\mathbf{w}} = \mathbf{p} - \mathbf{a}$$

$$v_\perp = [-v_y, v_x]^T$$

$$v_\perp . \vec{\mathbf{w}} > 0$$



Ou:

$$orient < \mathbf{a}, \mathbf{b}, \mathbf{p} >= \begin{vmatrix} 1 & a_x & a_y \\ 1 & b_x & b_y \\ 1 & p_x & p_y \end{vmatrix} > 0$$

Ou:

$$\vec{\mathbf{u}} = \vec{\mathbf{v}} \times \vec{\mathbf{w}}$$

$$u_z > 0$$

Localização de ponto em relação à reta

Ponto sobre L: caso degenerado

$$v_{\perp}.\vec{\mathbf{w}} = 0$$

Qual das 5 regiões?

Localização de ponto em relação à reta

Ponto sobre L: caso degenerado

$$v_{\perp}.\vec{\mathbf{w}} = 0$$

Qual das 5 regiões?

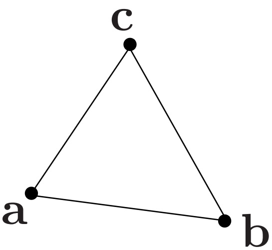
- Se $|v_x| > |v_y|$:
  - $p_x < a_x$: antes de $\mathbf{a}$
  - $p_x = a_x$: em $\mathbf{a}$
  - $a_x < p_x < b_x$: entre $\mathbf{a}$ e $\mathbf{b}$
  - $p_x = b_x$: em $\mathbf{b}$
  - $p_x > b_x$: depois de $\mathbf{b}$
- Se $|v_y| > |v_x|$:
  - $p_y < a_y$: antes de $\mathbf{a}$
  - $p_y = a_y$: em $\mathbf{a}$
  - $a_y < p_y < b_y$: entre $\mathbf{a}$ e $\mathbf{b}$
  - $p_y = b_y$: em $\mathbf{b}$
  - $p_y > b_y$: depois de $\mathbf{b}$

Localização de pontos

Localização de ponto em relação a um triângulo T definido pelos pontos $\mathbf{a}$, $\mathbf{b}$ e $\mathbf{c}$

Considere triângulo $T < \mathbf{a}, \mathbf{b}, \mathbf{c} >$

- Orientação positiva (anti-horária)

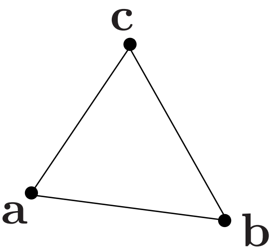


Localização de pontos

Localização de ponto em relação a um triângulo T definido pelos pontos $\mathbf{a}$, $\mathbf{b}$ e $\mathbf{c}$

Considere triângulo $T < \mathbf{a}, \mathbf{b}, \mathbf{c} >$

- Orientação positiva (anti-horária)



Uma solução:

- Classificar $\mathbf{p}$ em relação a cada uma das 3 retas

Localização de ponto em relação à triângulo $T < \mathbf{a}, \mathbf{b}, \mathbf{c} >$

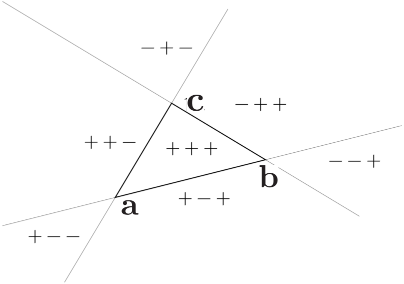
Solução alternativa:

- Determinar as coordenadas baricêntricas
  - Identifica região onde ponto se encontra

$$\mathbf{p} = \lambda_a \mathbf{a} + \lambda_b \mathbf{b} + \lambda_c \mathbf{c}$$

$$\lambda_a + \lambda_b + \lambda_c = 1$$

$$\lambda_a = \frac{\begin{vmatrix} p_x & b_x & c_x \\ p_y & b_y & c_y \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ 1 & 1 & 1 \end{vmatrix}}$$

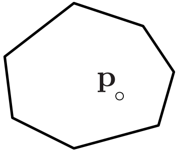


L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Localização de pontos

Localização de ponto em relação a um polígono convexo P

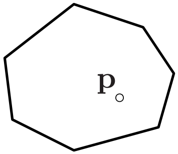
- Polígono convexo:
  interseção dos semi-planos definidos por suas arestas



Localização de pontos

Localização de ponto em relação a um polígono convexo P

- Polígono convexo:
  interseção dos semi-planos definidos por suas arestas



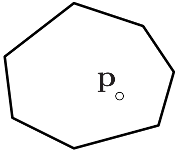
Uma solução:

- Classificar $\mathbf{p}$ em relação a cada uma das arestas
  - Tempo esperado: $O(n)$

Localização de pontos

Localização de ponto em relação a um polígono convexo P

- Polígono convexo:
  interseção dos semi-planos definidos por suas arestas



Uma solução:

- Classificar $\mathbf{p}$ em relação a cada uma das arestas
  - Tempo esperado: $O(n)$

Podemos melhorar o tempo esperado do algoritmo?

Localização de ponto em relação a um polígono convexo P

Explorando convexidade

- Considere o polígono convexo $P = \mathbf{p}_1\mathbf{p}_2...\mathbf{p}_n$
- Considere a reta $L = \mathbf{p}_1\mathbf{p}_k$, onde $k = \lfloor n/2 \rfloor$,
  dividindo o polígono em dois polígonos menores:

$$\begin{cases} P^- = \mathbf{p}_1\,\mathbf{p}_2\,...\,\mathbf{p}_k \\ P^+ = \mathbf{p}_1\,\mathbf{p}_k\,\mathbf{p}_{k+1}\,\mathbf{p}_{k+2}\,...\,\mathbf{p}_n \end{cases}$$

Localização de ponto em relação a um polígono convexo P

Explorando convexidade

- Considere o polígono convexo $P = \mathbf{p}_1\mathbf{p}_2 \ldots \mathbf{p}_n$
- Considere a reta $L = \mathbf{p}_1\mathbf{p}_k$, onde $k = \lfloor n/2 \rfloor$, dividindo o polígono em dois polígonos menores:

$$\begin{cases} P^- = \mathbf{p}_1\,\mathbf{p}_2 \ldots \mathbf{p}_k \\ P^+ = \mathbf{p}_1\,\mathbf{p}_k\,\mathbf{p}_{k+1}\,\mathbf{p}_{k+2} \ldots \mathbf{p}_n \end{cases}$$

- Se $\mathbf{p}$ estiver em L e entre $\mathbf{p}_1$ e $\mathbf{p}_k$: $\mathbf{p} \in P$
- Se $\mathbf{p}$ estiver em L e fora do segmento: $\mathbf{p} \notin P$
- Se $\mathbf{p}$ estiver à esquerda de L, localiza $\mathbf{p}$ em P^+
- Se $\mathbf{p}$ estiver à direita de L, localiza $\mathbf{p}$ em P^-

Localização de ponto em relação a um polígono convexo P

Explorando convexidade

- Considere o polígono convexo $P = \mathbf{p}_1\mathbf{p}_2 \ldots \mathbf{p}_n$
- Considere a reta $L = \mathbf{p}_1\mathbf{p}_k$, onde $k = \lfloor n/2 \rfloor$, dividindo o polígono em dois polígonos menores:

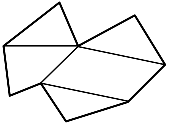
$$\begin{cases} P^- = \mathbf{p}_1\,\mathbf{p}_2 \ldots \mathbf{p}_k \\ P^+ = \mathbf{p}_1\,\mathbf{p}_k\,\mathbf{p}_{k+1}\,\mathbf{p}_{k+2} \ldots \mathbf{p}_n \end{cases}$$

  - Se $\mathbf{p}$ estiver em L e entre $\mathbf{p}_1$ e $\mathbf{p}_k$: $\mathbf{p} \in P$
  - Se $\mathbf{p}$ estiver em L e fora do segmento: $\mathbf{p} \notin P$
  - Se $\mathbf{p}$ estiver à esquerda de L, localiza $\mathbf{p}$ em P^+
  - Se $\mathbf{p}$ estiver à direita de L, localiza $\mathbf{p}$ em P^-
- Tempo esperado: $O(\log n)$

Localização de pontos

Localização de ponto em relação a uma subdivisão planar S

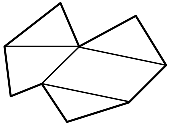
- Determinar se ponto pertence a uma face, a uma aresta, sobre um vértice ou fora

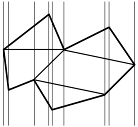


Localização de pontos

Localização de ponto em relação a uma subdivisão planar S

- Determinar se ponto pertence a uma face, a uma aresta, sobre um vértice ou fora





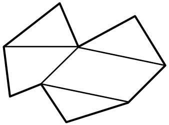
Solução: considere que não existem vértices com mesma abscissa x

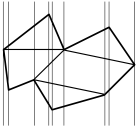
- Ordena vértices em x e traça retas verticais
  - Retas originam *fatias* na subdivisão planar
- Determinar fatia que contém $\mathbf{p}$

Localização de pontos

Localização de ponto em relação a uma subdivisão planar S

- Determinar se ponto pertence a uma face, a uma aresta, sobre um vértice ou fora

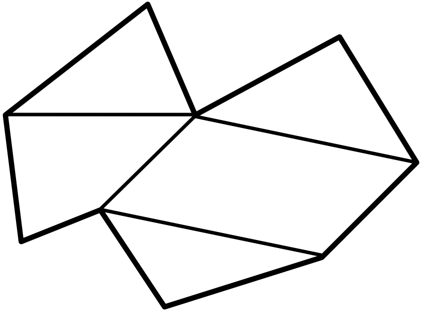


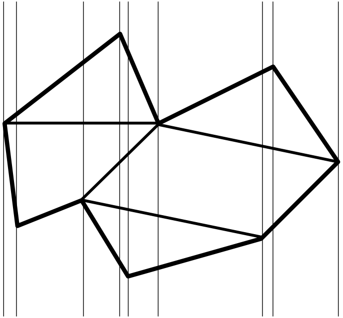


Solução: considere que não existem vértices com mesma abscissa x

- Ordena vértices em x e traça retas verticais : $O(n \log n)$
  - Retas originam *fatias* na subdivisão planar
- Determinar fatia que contém $\mathbf{p}$: $O(\log n)$

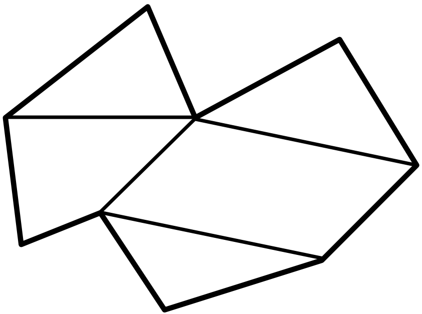
Localização de ponto em uma subdivisão planar S

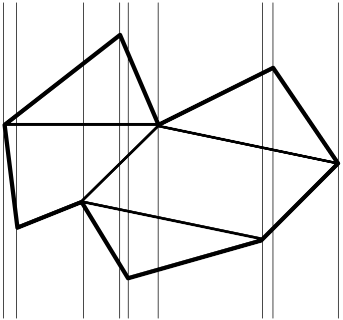




- Arestas dentro de fatias não se cruzam
  - Trapézios/triângulos podem ser ordenados em y
- Determinar trapézio/triângulo que contém $\mathbf{p}$

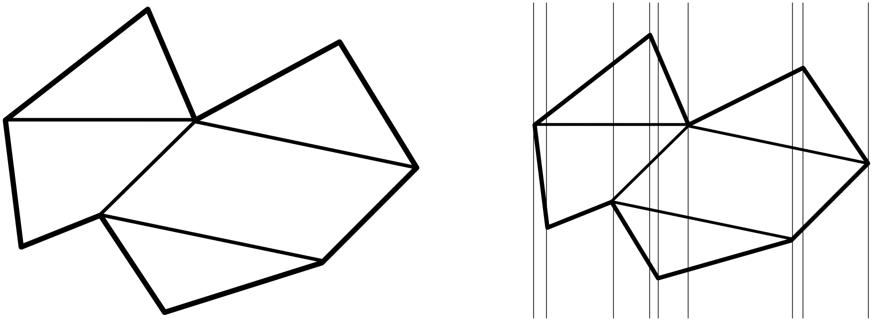
Localização de ponto em uma subdivisão planar S





- Arestas dentro de fatias não se cruzam
  - Trapézios/triângulos podem ser ordenados em y
- Determinar trapézio/triângulo que contém $\mathbf{p}$: $O(\log n)$

Localização de ponto em uma subdivisão planar S

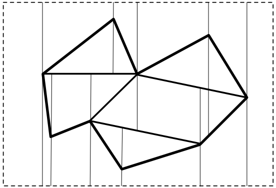


- Arestas dentro de fatias não se cruzam
  - Trapézios/triângulos podem ser ordenados em y
- Determinar trapézio/triângulo que contém $\mathbf{p}$: $O(\log n)$

Memória requerida: $O(n^2)$

Localização de ponto em uma subdivisão planar S

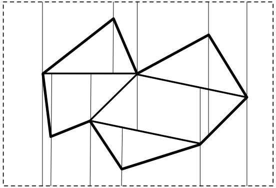
Mapa trapezoidal



- Cria retângulo envolvente para eliminar regiões ilimitadas
- De cada vértice, traça duas semi-retas: para baixo e para cima
  - Semi-retas interrompidas até interceptar uma aresta
- Número de trapézios (ou triângulos) nas fatias limitado

Localização de ponto em uma subdivisão planar S

Mapa trapezoidal

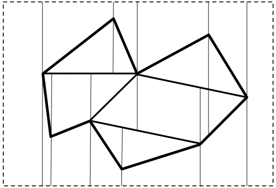


Propriedades de mapa trapezoidal

- Número de vértices: $\#v \leq 6n + 4$
- Número de trapézios: $\#e \leq 3n + 1$

Localização de ponto em uma subdivisão planar S

Mapa trapezoidal



Propriedades de mapa trapezoidal

- Número de vértices: $\#v \leq 6n + 4$
- Número de trapézios: $\#e \leq 3n + 1$

Memória requerida: $O(n)$

Par próximo

Determinar par mais próximo de um conjunto de pontos

Problema: Dado um conjunto de pontos P, determinar o par mais próximo

Determinar par mais próximo de um conjunto de pontos

Problema: Dado um conjunto de pontos P, determinar o par mais próximo

- Algoritmo força bruta: $O(n^2)$

Determinar par mais próximo de um conjunto de pontos

Problema: Dado um conjunto de pontos P, determinar o par mais próximo

- Algoritmo força bruta: $O(n^2)$

Como reduzir o esforço computacional?

Determinar par mais próximo de um conjunto de pontos

Problema: Dado um conjunto de pontos P, determinar o par mais próximo

- Algoritmo força bruta: $O(n^2)$

Como reduzir o esforço computacional?

- Se calcularmos a menor distância δ considerando $\mathbf{p}_1 ... \mathbf{p}_{k-1}$, ao considerar $\mathbf{p}_k$, temos duas situações:
  - $|\mathbf{p}_k \mathbf{p}_i| < \delta$, ou
  - O par mais próximo não se altera

Determinar par mais próximo de um conjunto de pontos

Problema: Dado um conjunto de pontos P, determinar o par mais próximo

- Algoritmo força bruta: $O(n^2)$

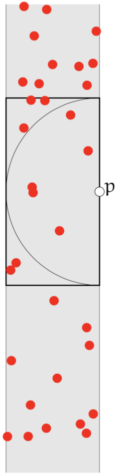
Como reduzir o esforço computacional?

- Se calcularmos a menor distância δ considerando $\mathbf{p}_1 ... \mathbf{p}_{k-1}$, ao considerar $\mathbf{p}_k$, temos duas situações:
  - $|\mathbf{p}_k \mathbf{p}_i| < \delta$, ou
  - O par mais próximo não se altera
- Então: só precisamos considerar pontos a uma distância menor que δ de $\mathbf{p}_k$.

Determinar par mais próximo de um conjunto de pontos

Algoritmo de varredura

- Manter lista L de candidatos, ordenados em y
- Ordenar pontos em x
  - Só precisamos verificar pontos $x_k - x_i < \delta$
- Inicializar varredura
  - $\delta = d(\mathbf{p}_1, \mathbf{p}_2)$, faixa $[x_1, x_2]$ e $L = \{\mathbf{p}_1, \mathbf{p}_2\}$
- Processar $\mathbf{p}_k$
  - Atualizar faixa para x_k
  - Remover candidatos da faixa: $x_k - x_i > \delta$
  - Acrescentar $\mathbf{p}_k$ à faixa, em ordem de y
  - Verificar distância de pontos em L vizinhos a $\mathbf{p}_k$



Determinar par mais próximo de um conjunto de pontos

Tempo esperado

1. Ordenar pontos em x
2. Inicializar varredura
3. Processar $\mathbf{p}_k$ ($n-2$ vezes)
   - Atualizar faixa para x_k
   - Remover candidatos da faixa: $x_k - x_i > \delta$
   - Acrescentar $\mathbf{p}_k$ à faixa, em ordem de y
   - Verificar distância de pontos em L a $\mathbf{p}_k$

L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Determinar par mais próximo de um conjunto de pontos

Tempo esperado

1. Ordenar pontos em x
2. Inicializar varredura
3. Processar $\mathbf{p}_k$ ($n-2$ vezes)
   - Atualizar faixa para x_k
   - Remover candidatos da faixa: $x_k - x_i > \delta$
   - Acrescentar $\mathbf{p}_k$ à faixa, em ordem de y
   - Verificar distância de pontos em L a $\mathbf{p}_k$

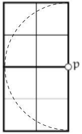
Tempo esperado?

L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Determinar par mais próximo de um conjunto de pontos

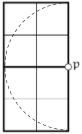
Tempo esperado

1. Ordenar pontos em x: $O(n \log n)$
2. Inicializar varredura: $O(1)$
3. Processar $\mathbf{p}_k$ ($n - 2$ vezes)
   - Atualizar faixa para x_k: $O(1)$
   - Remover candidatos da faixa: $x_k - x_i > \delta$: $O(1)$ (amortizado)
   - Acrescentar $\mathbf{p}_k$ à faixa, em ordem de y: $O(\log n)$
   - Verificar distância de pontos em L a $\mathbf{p}_k$: $O(1)$



L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Determinar par mais próximo de um conjunto de pontos



Tempo esperado

1. Ordenar pontos em x: $O(n \log n)$
2. Inicializar varredura: $O(1)$
3. Processar $\mathbf{p}_k$ ($n-2$ vezes)
   - Atualizar faixa para x_k: $O(1)$
   - Remover candidatos da faixa: $x_k - x_i > \delta$: $O(1)$ (amortizado)
   - Acrescentar $\mathbf{p}_k$ à faixa, em ordem de y: $O(\log n)$
   - Verificar distância de pontos em L a $\mathbf{p}_k$: $O(1)$

Tempo esperado total

- $O(n \log n)$: tempo ótimo
  - No máximo 8 candidatos para verificar distância a cada avanço de L

L.H. de Figueiredo, P.C.P. Carvalho, "Notas de Geometria Computacional", IMPA

Exercício

Considere dois conjuntos aleatórios de pontos P e Q no plano, com m e n elementos, respectivamente. É possível usar um algoritmo de varredura, baseado no de par mais próximo de um conjunto, para determinar o par $\{\mathbf{p}_i, \mathbf{q}_j\}$ mais próximo? Se sim, como seria seu pseudo-código e qual a ordem do tempo esperado para execução do seu algoritmo? É possível afirmar que o algoritmo proposto tem ordem de tempo ótimo?